BEST AVAILABLE COPY

PATENT SPECIFICATION (11) 1 247 746

1H

1 247 746

DRAWINGS ATTACHED

(21) Application No. 48405/69 (22) Filed 2 Oct. 1969
(31) Convention Application No. 771 791
(32) Filed 30 Oct. 1968 in
(33) United States of America (US)
(45) Complete Specification published 29 Sept. 1971
(51) International Classification G 06 f 11/00
(52) Index at acceptance
G4A 12D 12N 12P 15A2 16E2 16EY 16F 16J 17B2 17P 1F 2AY 2BY 2C 2F10 2F6 2F8 2G 3W10Y 3W2 6M1

(54) DATA PROCESSING MACHINES

(71) We, INTERNATIONAL BUSINESS MACHINES CORPORATION, a Corporation organized and existing under the laws of the State of New York in the United States of America, of Armonk, New York 10504, United States of America (assignees of Harold Fowler Heath Jr. and Samir Said Husson) do hereby declare the invention for which we pray that a patent may be granted to us, and the method by which it is to be performed, to be particularly described in and by the following statement:—

This invention relates to data processing machines.

The present invention provides a data processing machine including a plurality of functional units capable of cooperating in a plurality of configurations in order to execute a program, control means for determining from an examination of the current control word for each machine cycle the appropriate configuration of functional units to be operational (involved in the execution of the program), test means for inserting, during a machine cycle, a test pattern into a non-operational functional unit, error detecting means for detecting malfunction of the tested functional unit in relation to the test pattern and forcing means responsive to the error detecting means for forcing a predetermined routine on the machine on detection of a malfunction.

Considering the large number of components used in a computer, the reliability thereof becomes a problem. The large number of errors which can occur in a computer have been classified generally as either "solid" errors or "transient" errors. The "solid" error usually occurs because of the failure of one of the components in the system, whereas the "transient" error is one that may be intermittent such as might be caused by noise or other transient environmental conditions. Various schemes of error detection and correction have been devised and utilized in connection with computers. Probably the most widely known is the parity checking scheme which basically provides a number of bits which should have a predetermined value unless an error has occurred. Accordingly, if a parity check is made on data before it enters a particular functional unit and the check is again made at the output of the functional unit, it can be determined whether the error was introduced by the functional unit. In response to the error indication from a parity check, various procedures can be set in motion. For example, retry may be initiated which consists of sending the data back through the same functional unit to determine whether the error occurs again. If the error continues to occur as the functional unit is retryed, then the error is considered a "solid" error, as previously mentioned, and an error routine is entered. However, if on retry the error has disappeared, then the computation within the computer continues as the error is considered to be transient. It will be appreciated, that this type of error detecting indicates an error only after the computational data has been mutilated by the malfunctioning functional unit. Accordingly, if the error is determined to be "solid", it is necessary to go back to either the beginning and restart the job in the computer or to go back to a previously determined check point. These are points where the data is read out into auxiliary storage means where it is stored so that the computation can be returned to this point for restart once the unit giving rise to the failure has been fixed.

It would be very advantageous to discover a failed functional unit before it introduces an error into the data being processed. The present invention provides a means for discovering the failure of non-operational units before the error affects data. It has been found by observation that during any given

[Price 25p]

machine cycle the functional units utilized in the computation average less than half the functional units available. Accordingly, an average of over half the functional units are not being utilized in any given machine cycle. A system is provided operable within a computer for determining whether any functional unit is operational or non-operational during the given cycle. If the functional unit is found to be operational, and an error occurs, an error routine is entered in the usual manner. However, if the functional unit is found to be non-operational, then a test word is utilized in the functional unit to determine whether an error has been introduced. If an error has been introduced, a wait cycle is initiated to again pass the test data through the functional unit to determine whether the error occurs again. If the error re-occurs, the system enters the error routine. However, if the retry does not indicate an error, the wait cycle is cancelled and the computation allowed to continue. Since the testing is performed when the functional unit is non-operational, the error is discovered before the computational data is introduced so that the error is detected before the data is mutilated.

There are many advantages in detecting a fault in a functional unit before the unit can affect the computational data of the problem in the computer. If it is found that a "solid" error occurred, the computational data can be stored in an auxiliary storage facility until the failed functional unit is repaired. Thus, a check point need only be established if a "solid" error occurs. This results in establishing fewer check points and does not require any back up to the closest preceding check point. Early detection of the failure or error would also be advantageous in connection with reconfiguration schemes. It will be appreciated, that the reconfiguration could take place before the computational data is mutilated. There are also various other schemes for continuing the operation of a data processing system in spite of a malfunction. In one such system it is necessary to have auxiliary registers in which the information is stored so that the correct information is available for processing serially through an operational part of the functional unit when the other parts of the unit have failed. The data processing system continues to operate despite the malfunction. Thus, discovering the error by the early error detection means of the present invention would reduce the need for storing the information in the auxiliary registers.

The one embodiment of the invention comprises a digital computer having a plurality of functional units which are available for operation simultaneously in any given machine cycle. Means are provided for determining whether a functional unit is operational or non-operational during any machine cycle. A test word is introduced into the functional unit in response to a non-operational determination to determine the operational integrity of the functional unit. Each functional unit has an error indicating means associated therewith to indicate the occurrence of an error in the test data indicative of a failure in the associated functional unit. A predetermined computer routine is introduced in response to the error indication.

The present invention will be described further by way of example with reference to one embodiment of the invention, as illustrated in the accompanying drawings in which:—

FIGURE 1 is a schematic block diagram of a data processing machine or system according to the invention;

FIGURE 2 is a diagram of the general organisation of the sequence controls of the central processing unit of the system of Figure 1;

FIGURE 3 is a time chart of the timing circuit 306 shown in Figure 2;

FIGURE 4 is a schematic block diagram showing the invention operable in connection with the adder functional unit shown in Figure 1;

FIGURE 5 is a timing chart showing the timing for the operation of the apparatus shown in Figure 4; and

FIGURE 6 is a flow chart illustrating the steps taken in conjunction with the apparatus shown in Figure 4.

Basic Environmental System

The invention will be described and shown in the environment of an electronic digital computer containing a read only control storage which controls execution of program instructions. The invention is not limited to this type system but may be used in data processing machines which do not utilize a read only control storage, and in "special purpose computers" which are built specifically to perform only one (or a very limited number of) tasks, and which have a "program" built into the hardware of the machine.

The data processing system in which the present invention will be described typically includes storage, a central processing unit (CPU), a system control unit and some form of input/output (I/O) unit. Such a system is described in the following references for example:—

1. "IBM System/360 Principles of Operation" Form A22-6821;
2. "System/360 Model 50, Comprehensive Introduction" Form 223-2821;
3. "Microprogramming Manual for the IBM System/360 Model 50" by S. S. Husson, October 2, 1967, IBM Technical Report, TR 00. 1479-1; and

4. "Microprogram Control for System/360" by S. G. Tucker, IBM Systems Journal, Volume 6, Number 4, 1967, pages 222-241.

(IBM is a Registered Trade Mark).

With reference to Figure 1, the system storage includes main storage (MS) 12 and local storage (LS) 13. Although no input/output units are shown, such units are well known and communicate with the Figure 1 system through the gating network 216 into the adder output bus (AOB) latches 217 onto the (AOB) 221. The system control unit 11 controls the system operation by opening and closing gates and establishing other control signals at extensive locations throughout the system. Since such gating and control signals and their implementation are well known, they are collectively represented by the output bus 15. Specific control signals important to the present invention will be discussed further hereinafter. The remainder of the circuitry shown in Figure 1 is generally considered part of the CPU. The CPU and the system have the capability of executing store-in-place instructions.

Main Store.

The main storage (MS) 12 may be physically integrated with the CPU or constructed as a stand-alone unit. The storage cycle speed is not directly related to the internal cycling of the CPU, thereby permitting an efficient relationship of CPU speed to storage size. Fetching and storage of data by the CPU are not affected by any concurrent I/O data transfer.

The main store 12 is preferably a matrix array of magnetic cores where a given address in the array is selected by signals in the storage address register (SAR) 90. When the SAR 90 contains a main store address, the main store 12, under its own internal timing controls, operates through its basic memory cycle to read information onto output sense lines 95 into the storage data register (SDR) 91. From SDR 91, data may be regenerated back into MS 12 and sent through the gating circuitry 216, the AOB latches 217, onto the adder output bus (AOB) 221.

The basic memory cycle includes a read half cycle in which data are destructively read out from main storage into the SDR followed by a write half cycle in which the information in the SDR is regenerated back into main storage. By placing different information into the SDR 91 prior to regeneration on the write cycle, the information that was in main storage may be effectively changed. Simultaneously with the regeneration cycle, the information in the SDR 91 becomes available to the system on the AOB 221.

The information format of the environmental system organizes 8 bits into a basic building block called a "byte". Each byte also includes a ninth bit for parity used in error detection. The parity bit cannot be affected by the program, its only purpose being to cause a system interruption when a parity error occurs. It is assumed that the parity bit will be associated with bytes and that the normal parity checking circuitry is included throughout the system in the well known manner.

Two bytes are organized into a large field defined as a half-word, and four bytes or two half-words are organized into a still larger field called a word. More specifically, a "word" is defined as four consecutive bytes in the environmental system. However, it will be understood that words or bytes can equal any number of bits.

Various data formats may be employed in the environmental system so that instructions and operands may be of different lengths depending upon the particular operation which is to be carried out.

Bytes are assigned locations in storage in consecutively numbered positions starting with zero. Each number is considered the address of the corresponding byte. A group of bytes in storage is addressed by the leftmost byte of the group. The number of bytes in the group is either implicitly or explicitly defined by the operation specified by the instruction. The addressing arrangement uses a 24-bit binary address to accommodate a maximum of 16.777.216 byte addresses. This set of main storage addresses includes some locations dedicated to particular functions.

Storage addressing wraps around from the maximum byte address to the zero address. Variable-length operands may be located partially in the last and partially in the first location of storage, and are processed without any indication of crossing the maximum address boundary.

Fixed-length fields, such as half-words and double-words, must be located in main storage on an integral boundary for that unit of information.

A boundary is called integral for a unit of information when its storage address is a multiple of the length of the unit in bytes. For example, words (4 bytes) must be located in storage so that their address is a multiple of the number 4. Variable-length fields are not limited to integral boundaries, and may start on any byte location.

Local Store

Local store (LS) 13 consists of 64 one word capacity registers which are addressed by the local store address register (LSAR) 120. The LSAR 120 is loaded from the J register (J REG) 121 which is in turn fed from the AOB 221 or the mover out bus (MOB) 222. Whenever a read operation is specified from LS 13, the addressed word in LS 13 is read

out either to the L register (L REG) 126 or to the R register (R REG) 124. The L and R registers can have their outputs gated either back to the LS 13 or to the adder 210.

Local store 13 has a READ and WRITE operation similar to that of the main store 12.

Sixteen of the 64 one word locations in LS 13 are designated as general registers which are used as index registers in address arithmetic and indexing, and used as accumulators in fixed-point arithmetic and logical operations. These general registers are identified by numbers 0-15 and are specified by a 4-bit field in instructions. Additionally, LS 13 includes working store (WS) locations which are used for various purposes throughout processing.

Central Processing Unit (CPU)

There are three basic data-bus lines that are different in width, and through which data is channeled from one register to another. These are the 32-bit adder-out bus (AOB) 221, the 24-bit instruction-address bus (IAB) 223, and the 8-bit mover-out bus (MOB) 222.

The basic environmental system data flow consists primarily of two parallel paths which may be activated simultaneously. One is the 32-bit wide adder path including the adder 210 which is fed by the several 32-bit registers L, R, M and H. The other path is the 8-bit wide logical mover path including the 8-bit mover 213 fed by the L, R and M registers. The mover manipulates one-byte blocks in half-byte increments.

In addition to the adder and mover data paths, four other data paths are of interest in describing the basic environmental system; the shifter, instruction address, local storage, and main storage data paths.

The adder is capable of performing both binary and decimal arithmetic. Decimal arithmetic is performed by doing a binary add (true or complement) and generating a decimal correction factor into the L register in the same CPU cycle. Another cycle is needed to subtract the correction factor from the results of the preceding cycle. The adder 210 includes, besides 32 individual adder units, four parity checking circuits (one for each byte), four parity generating circuits (one for each byte), as well as carry look-ahead circuitry. When performing arithmetic functions, data are gated to the right-adder input Y from the 32-bit register H, M, or R. The left adder input XG contains a true/complement gate 220 and is fed by the 32-bit L register 126.

In a single CPU cycle, two 32-bit operands are gated one each into the XG and Y adder inputs, passed through the adder and continue on to set the adder output latches 217. At the end of the CPU cycle, the adder output is in the latches 217 ready to be gated out into an operating register. In the basic environmental system, subtraction is achieved by use of the two's complement which is controlled by the true/complement gate 220 on the XG input. When the complement gate is set, bits gated into XG will be inverted (i.e., one's become zeros and zeros becomes ones), thus forming the one's complement of the original XG input. The two's complement is achieved by inserting a carry into the XG adder input. Multiplication and division are accomplished using the adder by taking successive additions and subtractions. The various gating and control signals necessary to carry out the adder functions described emanate from the system control unit 11 which will be described in more detail hereinafter.

The shifter data path runs from the adder 210 to the AOB latches 217 and enables the adder output to be shifted to the left or the right either one or four places. Additionally, the shifter 215 includes means not shown for saving and storing the overflow portions of any shifted data. Again, the shifter is controlled by the system control unit 11.

The mover data path is used primarily for the execution of variable-field-length (VFL) instructions. Two byte sources may be selected simultaneously for a logical operation by the mover. The left-mover input, U, may be a byte selected from the L or R register under control of one of the two byte counters LB 101 and MB 102 or a byte formed by the contents of the two four-bit registers MD 103 and F 104. The right mover input, V, is a byte selected from the M register 211 under control of either byte counter LB or MB. The mover, like the other data paths, is controlled by the system unit 11.

The instruction address data path is 24 bits wide for moving and updating the 24-bit instruction contained in the instruction address register 218. The first instruction address is initially set in the instruction address register (IAR) by the system control unit 11. Instruction addresses are gated from the IAR 218 to the instruction address counter and latches 219. The instruction address counter increments the instruction address by the appropriate number of bytes (6 bytes in the case of restore in place or SS instructions) and places that updated address in the IAR via the bus 226. The current instruction address, before updating, represents the location in the main store 12 of the current instruction to be executed and it is read into the storage address register (SAR) 90, gated to the main storage 12, and causes the addressed instruction to be read out into the storage data register (SDR) 91.

Instructions read out from main store 12 into the SDR pass through the gating circuitry 216 to the AOB latches 217. The sequence of gating out an instruction is called I-fetch and is broken down into first and second level I-fetch. During I-fetch, the instruction is read out and is used to set up the CPU and local store with various initial conditions prior to commencement of execution.

The system control unit 11 includes a sequence control unit 302, general purpose stats 303, a program status word (PSW) register 304, and error detection circuitry 305.

Sequence Controls

Reference is made next to FIGS. 2 and 3 which show the sequence controls for the data processing system. The sequence controls include a capacitor read only store (ROS) 300 of the type described in an article entitled "Read Only Memory" by C. E. Owen et al on pages 47 and 48 of the *IBM Technical Disclosure Bulletin*, Volume 5, No. 8, dated January 1963. The controls also include a mode latch 307, condition triggers 303, also known as STATS, and timing circuits 306. The timing circuits 306 produce five cyclic signals at the CPU frequency which are phased with respect to the zero time reference to each CPU cycle as shown in FIG. 3.

Data in the read only store is addressed by a twelve-bit selection register (ROAR) 308. Address signals for the ROAR may be taken from various sources including a portion of the output control information from the read only store data register (ROSDR) 310 in each CPU cycle to select one of the 2,816 ninety-bit control words which are used in the environmental system and to enter the same in the read only storage data register 310. Actually, a twelve-bit ROAR register is capable of addressing 4,096 discrete locations. Each word, known as a microinstruction, is transferred into the read only store data register 310 at SENSE STROBE time which occurs just prior to the start of the next CPU cycle, and it controls the operation of the central processing unit during the next cycle.

The state of the read only store address register 308 is determined prior to the Drive Array pulse (FIG. 3) and controls the state of the read only store data register 310 at the following SENSE STROBE time. Thus, each entry into the read only store address register 308 usually controls the activity of the CPU in the next consecutive CPU cycle following the entry.

Each entry into the ROAR is determined in one of several different ways by the inputs presented to gates 312 through a network of OR gates 314. Ordinarily the 12 bits presented to the OR network 314 are derived selectively through gates 316 from one or more sources including a segment of the ROSDR, output conditions registered by selected condition STATS 303 and selected program branching information (program instruction operation codes).

The preceding discussion has presumed that the mode latch 307 is set to CPU mode and that CPU operation has not been interrupted by any input-output (I/O) units. Requests from I/O units are recognized by receipt of a Routine Received (RTNE RCVD) signal. It may be seen from the inputs to the AND gate 331 in FIG. 2 that, if the CPU is in the CPU mode when a RTNE RCVD signal is received, the mode latch 307 is not set to the I/O mode until SET REG time of the cycle following the rise of RTNE RCVD. This permits the CPU to complete execution of the current microinstruction. If the CPU mode is up when the RTNE RCVD signal is received, the AND gate 333 is operated to provide an output level which is up, and this level inhibits the AND circuit 332, thereby suppressing the SENSE STROBE signal of sense gates 334 which normally supply input signals to the read only storage data register 310 from the read only store 300. This will permit the I/O request to be serviced.

Detailed Description of the Invention

The invention will be described in connection with the adder functional unit shown in FIG. 1 and described above. It will be appreciated, that the invention is not limited to the adder function but is applicable to any function within the computer. Actually, the functional unit does not necessarily have to change the data (such as an adder) but it may be a unit which does not affect the data passing through it, such as a register or a data bus. During any cycle of the machine, each function is under the control of a particular control word. This control word is shown in the format of a ROS controlled machine although no such constraint is required. The ROS controlled words are found in the ROSDR 310 shown in FIGS. 2 and 4. The fields of the control words in the ROSDR are represented as C(Fi), C(Fj) .... C(Fk). Only the functional and necessary mechanization for carrying out the invention is shown in connection with control field C(Fi). The various control lines necessary for the operation of the adder function unit are not shown in FIG. 4. An all zero control field configuration has been selected as the non-operational indicator. With respect to a machine cycle, we call Fi operational if Fi is required and non-operational if Fi is not required, where Fi represents the functional unit 210. During a non-operational cycle, standard practice requires that Fi remain quiescent.

The present invention exercises Fi with test data during its non-operational cycles in order to determine if it has already failed. The bit lines (Sil, ... SiN) from C(Fi) in 310 are OR'ed together in OR unit 410 whose output is inverted by inverter 412. Thus, Fi 210 may either be operational (line Sl 411 is up) or non-operational (S2 413 is up) during the present machine cycle. It will be appreciated that line Sl will be energized or up for any control field C(Fi) configuration that it not completely zero. Likewise, if the control field is all zero's, the inverter 412 will produce an output causing line S2 to be up indicating that the functional unit Fi 210 is non-operational during that cycle of the machine.

FIG. 5 shows a timing chart dividing each machine cycle into six time periods t0 through t5 between which time periods control signals T0 — T5 are produced. During the occurrence of signal T0, AND circuit 424 is energized to produce an output signal S18 if the input signal S17 is present. The signal S18 inhibits the system from executing the next cycle by entering an error routine. The control field bit pattern C(Fi) is set at time T1.

Assuming that the control field C(Fi) does not contain the all zero pattern representing a non-operational unit, then the output line 411 of OR circuit 410 has a signal S1 thereon during time period T1 representing an operational condition of the functional unit 210. During this same time period T1, the solid error indicator 414 is reset. The functional unit 210 has completed its function at the end of period T2. If there is an error in the functional unit 210, its error indicator 416 is energized producing an output signal S3 on line 417 prior to t3. Output signal S3 forms one input to AND circuit 419. The arrival of time pulse T3 on input line 418 of AND circuit 419 along with signal S1 on line 411 causes an output S5 to be produced at the output of AND circuit 419. Output signal S5 sets the solid error indicator 414. Signal S5 is also connected to OR circuit 420 via line 421. The input of signal S5 to OR circuit 420 produces signal S13 which is applied to error indicator circuit 422. Signals S13 set error indicator circuit 422 so that it produces an output signal S17 which is connected to AND circuit 424. As previously mentioned, at time To, AND circuit 424 is energized so that an output signal S18 is produced which initiates the error routine. Of course, if the error indicator 416 indicates that there is no error, the result of the functional computation of functional unit 210 is fed out in the usual manner and error indicator 422 is not energized.

Assuming that the control field C(Fi) is all zeros, a non-operational indication is provided by the existence of signal S2 on line 413 as a result of the output from inverter 412. Since functional unit 210 is non-operational as indicated by the condition of line 413, then it is desired to gate test data into the inputs X and Y of functional unit 210. The appropriate test data is located in test data registers Z and D. These registers can be located anywhere in the system. The test patterns contained in the registers Z and D are dependent on both the function to be tested ($\overline{Fi}$) and its past history. The outputs Zo and Do are gated from the registers Z and D respectively, through AND circuits 430 and 431. The AND circuits 430 and 431 produce their respective output pulse S15 and S14 only when they receive, simultaneously, time pulse T2 and signal S2 from line 413 indicating a non-operational condition for the functional unit 210. The signals S15 and S14 are connected to the respective X and Y inputs of the functional unit 210 through OR circuits 433 and 432 respectively. It will be appreciated, that the test data will be gated to the functional unit 210 only when the function is indicated as being non-operational. If there is no failure in the functional unit 210 indicated by no output from the error indicator 416, then AND circuit 434 does not produce an output during the time period T3. Accordingly, transient error indicator 436 is not set and the first error counter 438 is not set. It will be noted that AND circuit 434 requires the simultaneous input of signal S2 representing a non-operational condition of the functional unit 210, signal S3 indicating an error in the functional unit and timing pulse T3 to produce an output signal S4 which sets the transient error indicator.

If an error occurs in the functional unit 210, a predetermined routine is introduced. An output signal S3 will be obtained from error indicator 416 which is fed to AND circuit 434 the output of which will set transient error indicator 436 at T3 time. If there was no error in the previous cycle of the machine, then first error counter 438 is in the reset condition. This can be seen by noting that AND circuit 440 has two inputs, one from the transient error indicator 436 when it is in the reset condition and the other from the CPU timing circuit 306 at time period T5. Thus, AND circuit 440 produces an output signal S7 which resets first error counter 438 only when the transient error indicator 436 is in its reset condition. Consequently, error indicator 422 is connected to the set output of first error counter 438 through AND circuit 442 and OR circuit 420. The other input to AND circuit 442 is the output of AND circuit 434 via line 444. Thus, AND circuit 442 will produce an output only when AND

circuit 434 produces an output and when first error counter 438 is in the set condition. The output of OR circuit 420 is utilized to set error indicator 422. However, the output of first error counter 438 in the reset condition serves as an input to AND circuit 446 which upon receiving time pulse T4 will produce an output if a simultaneous input signal is present from the set condition of transient error indicator 436. The output of AND circuit 446 passes through OR circuit 448 and sets the wait trigger 449 which produces an output signal S16 which is utilized to cause the machine to go into a wait cycle during which the normal operation of the machine is suspended and functional unit 210 is again tested for a malfunction to determine if the error was solid or intermittent. At T5 time of the same cycle, the output of AND circuit 451 will produce an output signal S8 when an input signal S12 is present as an input from the set condition of transient error indicator 436. This signal S8 is used to set first error counter 438. During the ensuing wait cycle, a test pattern is gated into the functional unit 210 during the T2 time period. If there is no error indicated by error indicator 416 during the wait cycle, then the previous error was obviously an intermittent error and can be ignored. Consequently, first error counter 438 is reset again at T5 time by the output of AND circuit 440, and the system returns to the operational program. However, if the functional unit 210 fails during the wait cycle, then an output is produced by error indicator 416 which forms an input to AND circuit 434 along with the input signal S2 indicating a non-operational condition of the functional unit and an input at T3 time. In response to these inputs, AND circuit 434 will produce an output on line 444 which forms an input to AND circuit 442 which already has another input from the set condition of the first error counter 438. Accordingly, AND circuit 442 will produce an output signal S10 which passes through OR circuit 420 to error indicator 422 which causes an error routine. The error routine is initiated since two successive errors have been indicated in the functional unit 210. The first error counter 438 can be a multistage counter rather than the two state device shown. Thus, a predetermined number of delays can be introduced during each of which the functional unit can be again tested to determine whether the error still occurs. The counter would indicate a reset condition for each state of the counter until the last stage when a set condition would allow the next S4 signal from AND circuit 434 to set error indicator 422 via AND circuit 442 and OR circuit 420.

The output of AND circuit 434 also drives a transient counter 450 through OR circuit 452. Whenever a failure occurs during a non-operational cycle in any of the functional units, the transient counter 450 is incremented by "1". The state of counter 450 is a measure of the operational reliability of the system. The transient counter 450 is essentially a warning device. The output of the counter 450 can be used to enter a test routine which will provide further information about the system reliability. Of course, there can be a separate transient counter for each function. This will permit a tighter control on system reliability.

The operation of the invention can be more clearly understood with reference to the timing chart of FIG. 5. The timing chart represents one complete machine cycle which is broken down into separate timing periods or pulses T0 through T5. However, in actuality, the machine time functions are not broken down into fixed time periods such as T0 through T5. The fixed time period arrangement is utilized for convenience of explanation only. In general, the functions of the system can be divided into two classes during every machine cycle; namely, the operational and the non-operational functions. The number of functional elements which are operational is clearly unimportant because the failure in any one of them must stop the machine due to the uncertain effect on the operational program and its data sets. This is accomplished as shown in FIG. 4. The outputs from the AND circuits 419 of the various functional units in the system are OR'ed together in OR circuit 420. The output of OR circuit 420 is used to set the error indicator 422. If one or more of the functions fail, then the corresponding solid error indicators 414 are set and the error routine can determine which function failed by interrogating all those indicators.

Referring to FIG. 6 which is a flow chart of operations, it can be seen that the new operation or cycle of the machine is entered at time T0. During this new cycle, if the error indicator 422 is on, the error routine is entered. If the error indicator 422 is not on, then at T1 time indicators 416, 414, and 436 are reset. It is also necessary during this time period T1 to determine if the wait trigger 449 is on. If the wait trigger 449 is on, it is turned off and new test data is gated to the functional unit 210 during the time T2. If the wait trigger 449 is not on, then it must be determined whether the functional unit 210 is to be operational or non-operational during the machine cycle. This is accomplished by determining whether the control pattern C(Fi) indicates an operational or non-operational pattern. If the functional unit 210 is to be operational, then the computation is completed in functional unit 210 during

time T2. If the error indicator 416 indicates a failure in functional unit 210, the error indicator 416 output causes error indicator 422 to turn on and an error routine will be entered at the beginning of the next cycle. If functional unit 210 did not fail, then the new cycle is entered into and the error indicator 422 is not on, therefore the operation proceeds as indicated above. If the functional unit 210 is to be non-operational during the machine cycle, then during time period T2 the test pattern stored in registers Z and D is gated to functional unit 210 where the operation on the data is completed. If functional unit 210 did not fail, then the first error counter 438 is reset at T5 time and a new cycle is again entered. If the functional unit 210 does fail, then the transient error indicator 436 is set at T3 time. If 210 failed last cycle, which is determined by the condition of first error counter 438, then the error indicator 422 is set and the error routine is entered at the beginning of the next cycle. If the functional unit 210 did not fail on the last cycle, then the wait trigger 449 is turned on at T4 time and the first error counter 438 is set at T5 time so that an error in functional unit 210 can set the error indicator 422. Thus, it will be appreciated that, if a non-operational pattern is obtained for a particular functional unit, test data can be gated to the unit to determine whether the unit has an error therein. If an error occurs, the arrangement is such that a wait cycle can be initiated during which the same functional unit is again tested to determine whether the error reoccurs. If the error reoccurs, then the error routine is entered. However, if the error does not reoccur, then the regular operational cycle is resumed. Using this detecting technique, the errors can be found before they actually affect the operational data. Consequently, various measures can be introduced to possibly prevent the shutdown of the machine because of a defective functional unit. For example, in a redundant system, the functional operation to be carried out by the defective unit can be transferred to a redundant functional unit.

WHAT WE CLAIM IS:--

1. A data processing machine including a plurality of functional units capable of cooperating in a plurality of configurations in order to execute a program, control means for determining from an examination of the current control word for each machine cycle the appropriate configuration of functional units to be operational (involved in the execution of the program), test means for inserting, during a machine cycle, a test pattern into a non-operational functional unit, error detecting means for detecting malfunction of the tested functional unit in relation to the test pattern and forcing means responsive to the error detecting means for forcing a predetermined routine on the machine on detection of a malfunction.

2. A machine as claimed in claim 1, wherein the control means includes a register into which a current control word is loaded and means for determining whether said control word in said register is a predetermined word indicative of a non-operational control word in respect of particular functional units.

3. A machine as claimed in claim 1 or claim 2 wherein the test means comprises one or more registers for holding said test pattern and gating means for gating said test pattern to said functional unit in response to a non-operational determination for the respective functional unit during a given cycle.

4. A machine as claimed in any preceding claim wherein counting means are provided for counting each error indication from said functional units which are non-operational during a given machine cycle to determine the reliability.

5. A machine as claimed in any preceding Claim wherein the forcing means comprises means for initiating a wait cycle during which the functional unit is again tested; means responsive to a further error indication for causing the machine to enter an error routine; and means responsive to a no error indication for resetting said means for initiating a wait cycle (if set).

6. A machine as claimed in Claim 5, wherein said wait cycle means is repeated for a predetermined plurality of successive error indications before said error routine means is initiated.

7. A machine as claimed in Claim 5, wherein means are provided for initiating said wait cycle in accordance with an error indication from any of a plurality of functional units as long as the error is the first error from that particular functional unit; and means for initiating said error routine in response to a successive error from the same functional unit.

8. A machine as claimed in Claim 5, wherein said means responsive to said error indication for initiating a wait cycle comprises a transient error indicator responsive to an error indication during a machine cycle in which the functional unit is non-operational; and a first error counter, the transient error indicator initiating a wait cycle when said first error counter is in the reset condition, the error routine being entered in response to a set condition of said first error counter during a non-operational cycle of said machine.

9. A machine as claimed in Claim 8 wherein said transient error indicator is effective in its reset condition to reset the

first error counter in response to a predetermined timing pulse so that a wait cycle can be entered in response to a subsequent error signal from said error indicator during a non-operational cycle of said machine.

10. A data processing machine substantially as hereinbefore described with reference to and as illustrated in the accompanying drawings.

I. M. GRANT,
Chartered Patent Agent,

Agent for the Applicants.

Printed for Her Majesty's Stationery Office by The Tweeddale Press Ltd., Berwick-upon-Tweed. 1971
Published at the Patent Office, 25 Southampton Buildings, London WC2A 1AY from which copies may be obtained.

BEST AVAILABLE COPY

1247746 COMPLETE SPECIFICATION

4 SHEETS *This drawing is a reproduction of the Original on a reduced scale*

Sheet 1

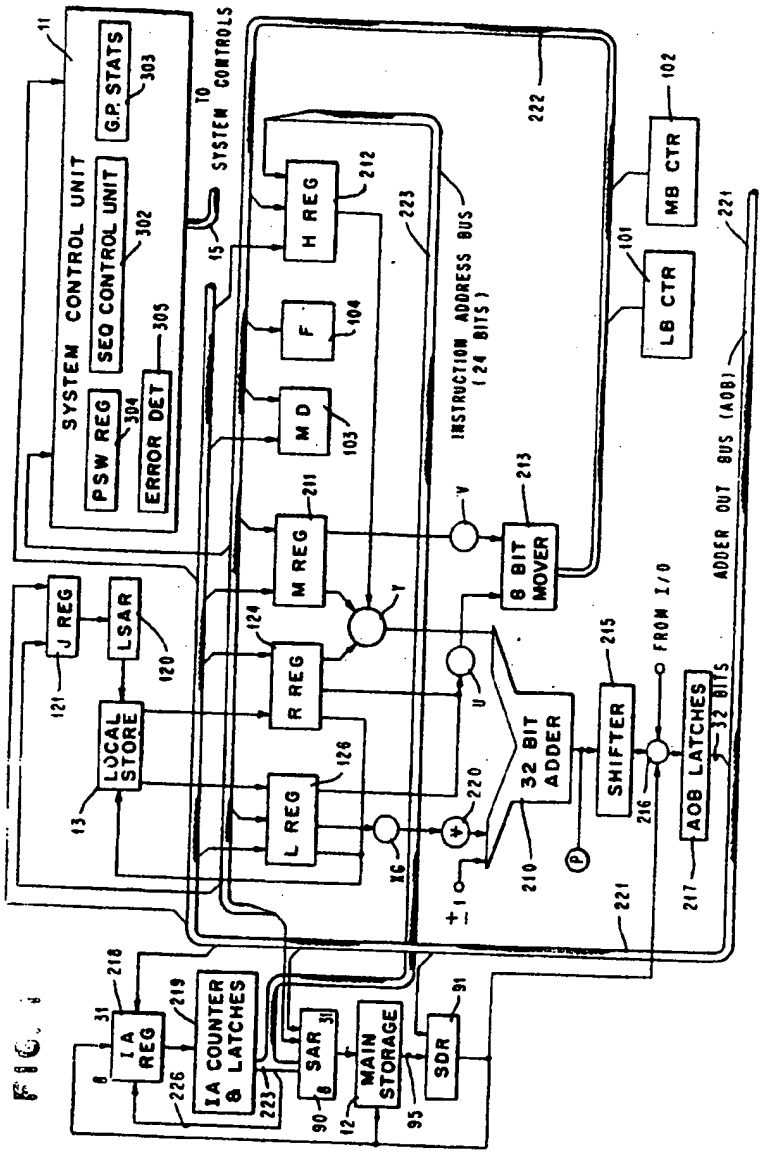


FIG. 1

BEST AVAILABLE COPY

1247746 COMPLETE SPECIFICATION

4 SHEETS *This drawing is a reproduction of the Original on a reduced scale*

Sheet 2

FIG. 2

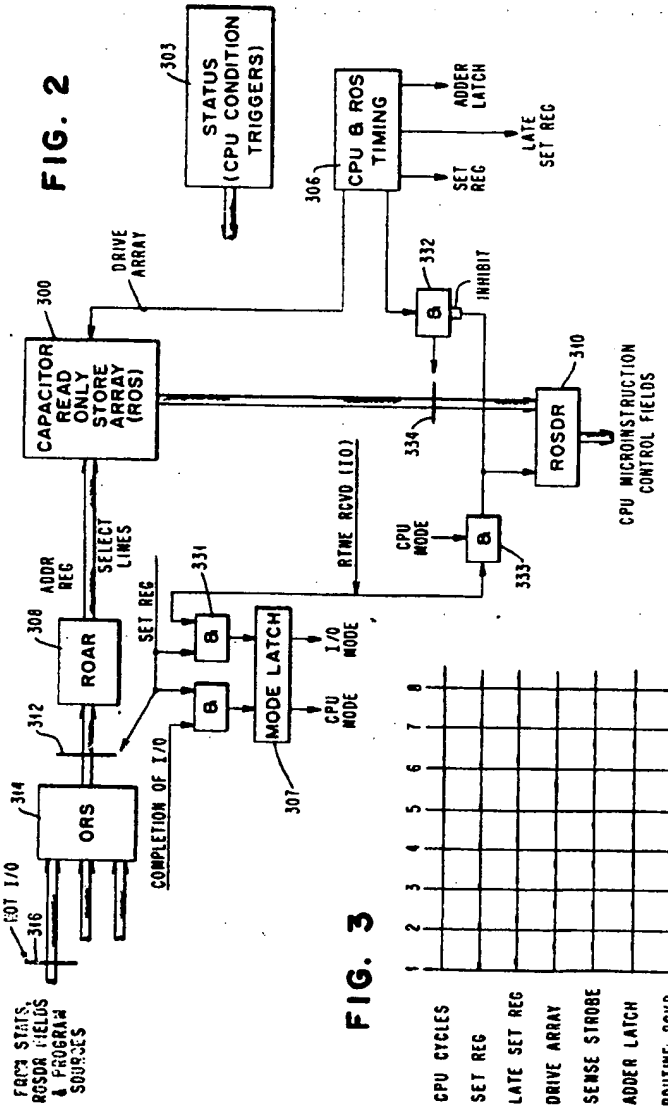
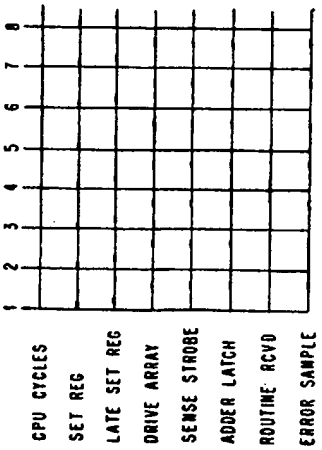


FIG. 3



BEST AVAILABLE COPY

1247746 COMPLETE SPECIFICATION

4 SHEETS This drawing is a reproduction of the Original on a reduced scale

Sheet 3

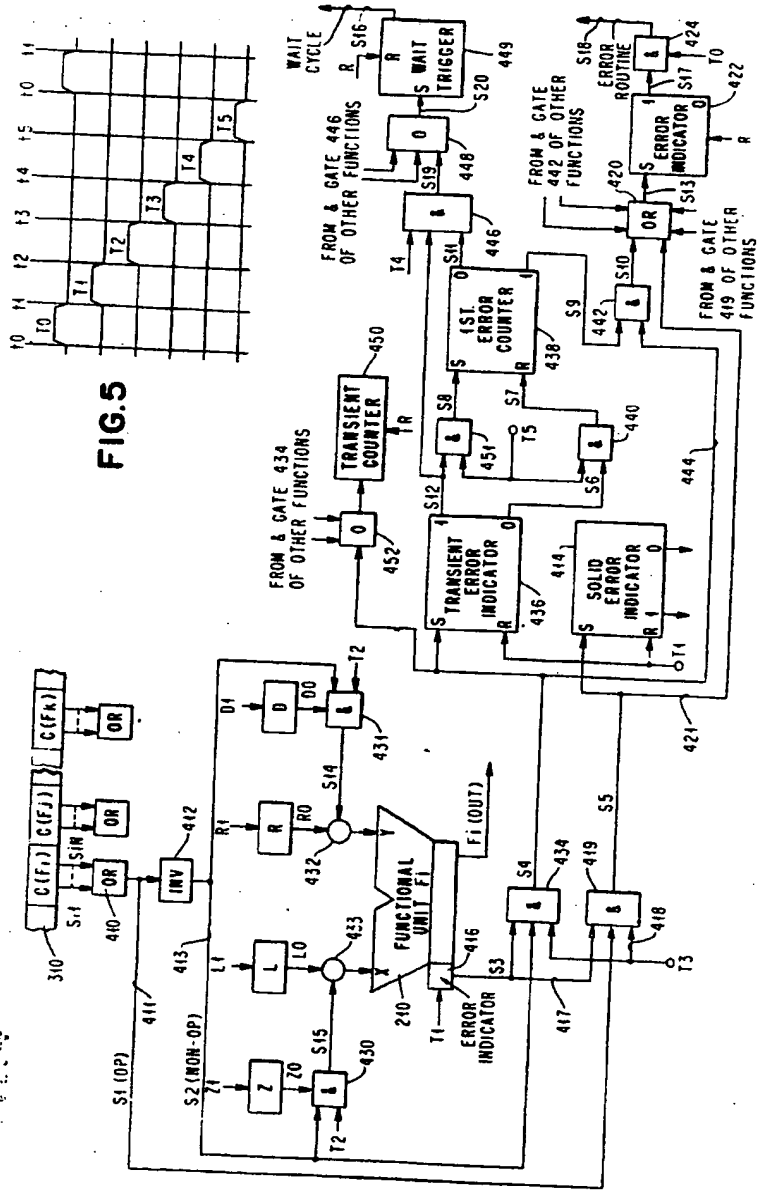


FIG.5

BEST AVAILABLE COPY

1247746 COMPLETE SPECIFICATION

4 SHEETS *This drawing is a reproduction of the Original on a reduced scale*

Sheet 4

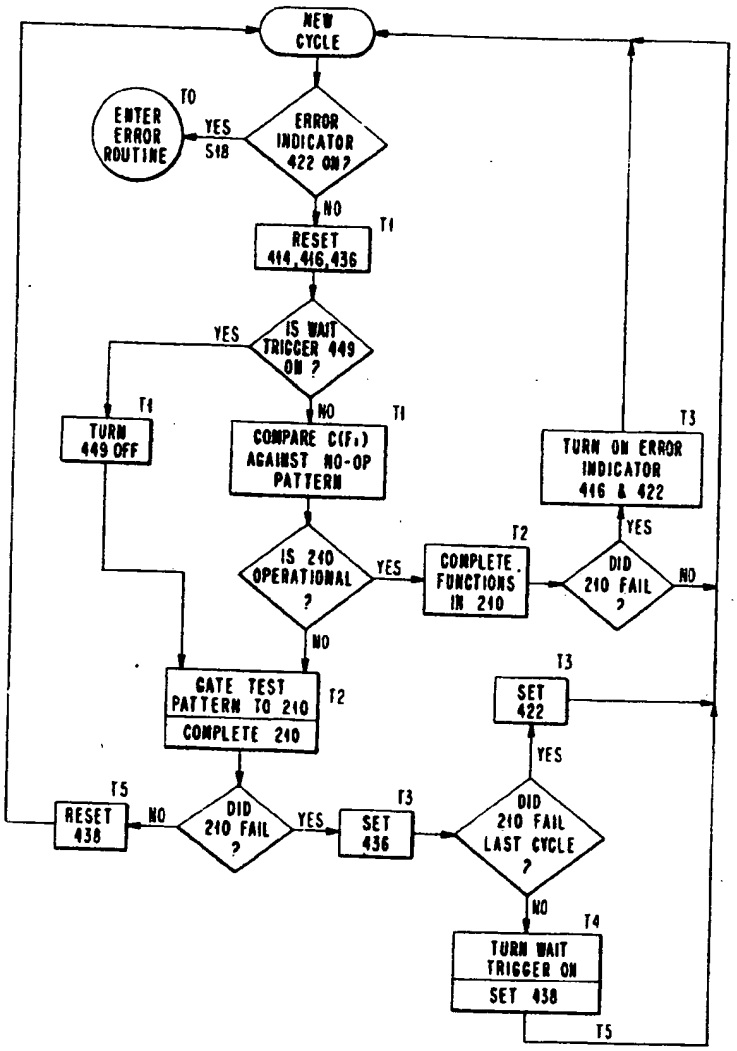



FIG. 6